Btrfs
Current Status and Future Prospects

Oct 13 2014
Satoru Takeuchi
Linux Development Div.
Fujitsu LTD.
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Background

- Fujitsu has developed Btrfs for Mission Critical (MC) systems since 2010

Requirements of MC systems

- High robustness
  - Don’t crash: data duplication
  - Error detection: checksum
  - Repair, recovery: snapshot, backup/restore, repairing tools

- High availability: Should work 365 days/24h
  - Limited maintenance time: enlarge storage size and backup online

- Btrfs is designed for such the requirements
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Core Features

- Multi-volumes
- Copy-on-Write Style Update
- Data/Metadata Checksum
- Subvolume
- Snapshot
- Transparent Compression
Multi-volumes

- Btrfs file system can consists of multiple volumes
  - Low layered and low overhead than LVM
  - Many features: RAID, online {add/remove/replace} devices

XFS or ext4 + LVM

<table>
<thead>
<tr>
<th>VFS</th>
<th>File system (XFS, ext4 and so on)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVM</td>
<td></td>
</tr>
<tr>
<td>Block device</td>
<td></td>
</tr>
</tbody>
</table>

Btrfs

<table>
<thead>
<tr>
<th>VFS</th>
<th>Btrfs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Block device

# mkfs.btrfs /dev/sd{a,b,c}1
Copy-on-Write (CoW) style update

- **Btrfs uses CoW style data/metadata update**
  - Safer than overwrite style update by design

- **Overwrite style: Update the data in place**

```
file
  ↓
data
file
  ↓
data
file
  ↓
data
```

System crash => data becomes inconsistent

- **CoW style: Copy, update, and replace pointer**

```
file
  ↓
data
file
  ↓
data
file
  ↓
data
```

System crash => data keeps consistency

Will be deleted later
CoW versus Overwrite

1,000 surprising power failure test

- Linux File System Analysis for IVI system, Mitsuharu Ito, Fujitsu

Result

- Ext4: Metadata was corrupted
- Btrfs: Worked fine without any problem

In my internal similar testing, XFS corrupted too.
Btrfs has checksum for each data/metadata extent to detect and repair the broken data.

When Btrfs reads a broken extent, it detects checksum inconsistency:
- With mirroring: RAID1/RAID10
  - Read a correct copy
  - Repair a broken extent with a correct copy
- Without mirroring
  - Dispose a broken extent and return EIO

With “btrfs scrub”, Btrfs traverses all extents and fix incorrect ones:
- Online background job
A subvolume is a file system inside file system

- Can be treated as a file system root
  - Mountable: most mount options are shared
  - Own inode namespace and quota limit

Efficient: Available space is shared

# btrfs subvolume create sub
**Snapshot**

- **Copy of a subvolume**
  - Far faster than LVM
    - Not a full copy, but only update metadata in CoW style
- **ReadOnly snapshot: with –r option**
- **Incremental snapshot: snapshot of snapshot**

```
# btrfs subvolume snapshot [-r] ./sub ./snap
```

Reference count

Capture a snapshot

Update data C in a snapshot
Performance of Snapshot: Btrfs versus LVM

1. Copy the following data to a volume
   - Consists of 100 directories and 100 files for each directory
     - File size: 1MB

2. Capture a snapshot of the volume

<table>
<thead>
<tr>
<th>Hardware Environment</th>
<th>Software Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>• PRIMERGY RX300 S6</td>
<td>• Red Hat Enterprise Linux 7.0</td>
</tr>
<tr>
<td>• CPU: Intel Xeon X5690 3.47GHz x12 core</td>
<td>• File systems</td>
</tr>
<tr>
<td>• Memory: 16GiB</td>
<td>• Btrfs</td>
</tr>
<tr>
<td>• Storages: 100GB HDD x 2</td>
<td>• Data/metadata: RAID1</td>
</tr>
<tr>
<td></td>
<td>• Other options: default</td>
</tr>
<tr>
<td></td>
<td>• XFS: default options</td>
</tr>
<tr>
<td></td>
<td>• Volume manager for XFS</td>
</tr>
<tr>
<td></td>
<td>• dm-thinp: chunksize is 256KiB</td>
</tr>
<tr>
<td></td>
<td>• LVM: RAID1</td>
</tr>
</tbody>
</table>
Result

- Copy: Btrfs  >  LVM  >>>>  dm-thinp
- Snapshot: Btrfs  >  dm-thinp >>>> LVM

<table>
<thead>
<tr>
<th>Volume type</th>
<th>Copy</th>
<th>Snapshot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Without page cache</td>
</tr>
<tr>
<td>Btrfs</td>
<td>106s</td>
<td>0.126s</td>
</tr>
<tr>
<td>XFS on dm-thinp</td>
<td>209s</td>
<td>0.260s</td>
</tr>
<tr>
<td>XFS on LVM</td>
<td>133s</td>
<td>1.03s</td>
</tr>
</tbody>
</table>
Transparent compression

- Automatically compress/expand file data on I/O
- Low space consumption and high I/O performance
  - Need some extra CPU time
- Usage: mount -o compress={lzo,zlib} <device> <mnt point>
  - Can also be enabled/disabled for each file

![Diagram showing system and storage with and without compression]
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Developments statistics

- Patch statistics
- Performance
- Summary
Patch Statistics

- bugfix
- functional enhancement
- performance enhancement
- cleanup

Copyright 2014 FUJITSU LIMITED
Patch Statistics: Tips of v3.17

Rejected by Linus
Patch statistics: Main changes

- **auto defrag scrub**
- **replace subcommand**
- **quota send/receive**
- **RAID5/6**
- **offline dedup**
- **btrfsck repair**
- **Improve sync write ~60%**
- **Improve error handling**
- **Inode properties**

Colors indicate:
- Blue: bugfix
- Red: functional enhancement
- Orange: performance enhancement
- Green: cleanup

Copyright 2014 FUJITSU LIMITED
Fujitsu’s contribution

- btrfsck, error handling
- fast {random/async} write
- LZO compression
- read only snapshot
- random Bug fixes
- enrich xfstests
Fujitsu’s contribution: btrfs-progs

- fsck
- error handling
- random bug fixes
- enrich xfstests
- documentation
## Performance measurement

<table>
<thead>
<tr>
<th>Hardware Environment</th>
<th>Software Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PRIMERGY TX300 S6</strong></td>
<td>Benchmark software: filebench</td>
</tr>
<tr>
<td>• CPU: Xeon x5670 x 2</td>
<td>• Kernel: 3.14.2, 3.15.3, 3.16.0, and 3.17-rc2</td>
</tr>
<tr>
<td>• 12 core</td>
<td>• I/O scheduler: deadline</td>
</tr>
<tr>
<td>• HT is disabled</td>
<td>• File systems: Btrfs(single volume), XFS, and ext4</td>
</tr>
<tr>
<td>• Memory: 4GB</td>
<td>• default mkfs options and mount options</td>
</tr>
<tr>
<td>• HDD: 300GB x 1</td>
<td></td>
</tr>
<tr>
<td>• MegaRAID SAS, HITACHI</td>
<td></td>
</tr>
<tr>
<td>HUS156030VLS600</td>
<td></td>
</tr>
</tbody>
</table>
The result: Compare with other file systems

Kernel version: v3.17-rc2
The result: Compare with old Btrfses
VFS has also improved performance

Accomplished by VFS layer performance enhancement
Summary

- Ready to use without RAID5/6
  - Performance: OK
  - Stability: OK
    - # of new features has decreased
    - Test coverage has increased
- Features: almost OK
  - RAID5/6: Lack of scrub and replace subcommands
- RAID1 and RAID10 are the best choice
  - Especially safe and stable
Agenda

- Background
- Core Features
- Developments statistics
- Future Prospects
Future Prospects: Fujitsu’s plan

■ RAID 5/6 enhancement
   ■ Add scrub and replace subcommands
     • We’re testing patches now and will post it to linux-btrfs ML soon
   ■ Add five tests for these features to xfstests

■ Further enhancement of robustness and performance
   ■ Repairing tools and so on

■ Education and documents for this purpose
   ■ Operation know-how
     • Btrfs operations are different from other file systems
       • e.g. Btrfsの基礎 part1 機能編 (It’s in Japanese. Now translating to English…)
         http://www.slideshare.net/fj_staoru_takeuchi/btrfs-part1

■ File system structure
■ Code logic
Future Prospects: Btrfs users are increasing

- OpenSuSE13.2: Will be used as its default
- Ubuntu: Support
- RHEL7: Available as tech-preview
- Will be used for In Vehicle Infotainment (IVI) system

Linux File System Analysis for IVI system, Mitsuharu Ito, Fujitsu

http://events.linuxfoundation.jp/sites/events/files/slides/linux_file_system_analysis_for_IVI_systems.pdf
Conclusion

■ Please try Btrfs
■ It’s ready to use
  ■ RAID1/10 are the best choice
  ■ RAID5/6 need some more work
  ■ Recommend the newest stable kernel
References

- Linux File System Analysis for IVI system, Mitsuharu Ito, Fujitsu
  http://events.linuxfoundation.jp/sites/events/files/slides/linux_file_system_analysis_for_IVI_systems.pdf

- Btrfsの基礎 part1 機能編
  http://www.slideshare.net/fj_stooru_takeuchi/btrfs-part1

- Linux-btrfs ML
  linux-btrfs@vger.kernel.org

- Btrfs wiki
  https://btrfs.wiki.kernel.org/index.php/Main_Page